call void @llvm.dbg.value(metadata ptr %func args, metadata !84, metadata ...!DIExpression()),!dbg!92 %call = tail call i32 @initialise arrays(ptr noundef nonnull @ func .s253) ... #8, !dbq !93 %call1 = tail call i32 @gettimeofday(ptr noundef %func_args, ptr noundef ... null) #8, !dbg !94 tail call void asm sideeffect ".inst 0x2520e020", ""() #8, !dbg !95, !srcloc call void @llvm.dbg.value(metadata i32 0, metadata !86, metadata ... !DIExpression()), !dbg !98 br label %for.cond2.preheader, !dbg !99 for.cond2.preheader: %nl.038 = phi i32 [0, %entry], [%inc21, %for.cond.cleanup4] call void @llvm.dbg.value(metadata i32 %nl.038, metadata !86, metadata ...!DIExpression()),!dbg!98 call void @llvm.dbg.value(metadata i32 0, metadata !88, metadata ...!DIExpression()),!dbg!100 %0 = call i64 @llvm.vscale.i64(), !dbg !101 %1 = mul i64 %0, 2, !dbg !101 %2 = icmp uge i64 8192, %1, !dbg !101 br i1 %2, label %pre.alc, label %Preheader.for.remaining.iterations, !dbg ...!101 Τ F pre.alc: %8 = call i64 @llvm.vscale.i64() %9 = mul i64 %8, 2 %uniform.vector = call <vscale x 2 x i64> ... @llvm.experimental.stepvector.nxv2i64() $%10 = \text{urem } i64\ 8192, \%9$ %total.iterations.to.be.vectorized = sub i64 8192, %10 %11 = insertelement <vscale x 2 x i64> poison, i64 %9, i64 0 %stepVector.update.values = shufflevector <vscale x 2 x i64> %11, <vscale x ... 2 x i64> poison, <vscale x 2 x i32> zeroinitializer %remaining.vector = add <vscale x 2 x i64> %uniform.vector, ... %stepVector.update.values %12 = getelementptr inbounds [8192 x i32], ptr @a, i64 0, i64 0, !dbg !116 %13 = getelementptr inbounds [8192 x i32], ptr @b, i64 0, i64 0, !dbg !124 %14 = call i64 @llvm.vscale.i64(), !dbg !116 %15 = mul i64 %14, 2, !dbg !116 %16 = getelementptr inbounds [8192 x i32], ptr @a, i64 0, i64 %15, !dbg !116 %17 = getelementptr inbounds [8192 x i32], ptr @b, i64 0, i64 %15, !dbg !124 %18 = load < vscale x 2 x i32 > , ptr %12, align 8%19 = load < vscale x 2 x i32 >, ptr %13, align 8%21 = load < vscale x 2 x i32 > , ptr %16, align 8%22 = load < vscale x 2 x i32 >, ptr %17, align 8 br label %alc.header alc.header: %24 = phi i64 [0, %pre.alc], [%52, %new.latch] %25 = phi < vscale x 2 x i64 > [%uniform.vector, %pre.alc], [%53,... %new.latch] %26 = phi <vscale x 2 x i64> [%remaining.vector, %pre.alc], [%54, ... %new.latch 1 br i1 true, label %lane.gather, label %linearized F lane.gather: %27 = call < vscale x 2 x i1 > @llvm.aarch64.sve.ptrue.nxv2i1(i32 2)%28 = call <vscale x 2 x i64> @llvm.aarch64.sve.compact.nxv2i64(<vscale x 2 ... x i1> %20, <vscale x 2 x i64> %uniform.vector) %29 = call <vscale x 2 x i64> @llvm.aarch64.sve.compact.nxv2i64(<vscale x 2 ... x i1 > %20, <vscale x 2 x i64 > %remaining.vector) %30 = xor < vscale x 2 x i1 > %20, shufflevector (< vscale x 2 x i1 >... insertelement (<vscale x 2 x i1> poison, i1 true, i32 0), <vscale x 2 x i1> ... poison, <vscale x 2 x i32> zeroinitializer) %31 = xor <vscale x 2 x i1> %20, shufflevector (<vscale x 2 x i1> ... insertelement (<vscale x 2 x i1> poison, i1 true, i32 0), <vscale x 2 x i1> ... poison, <vscale x 2 x i32> zeroinitializer) %32 = call <vscale x 2 x i64> @llvm.aarch64.sve.compact.nxv2i64(<vscale x 2 ... x i1> %30, <vscale x 2 x i64> %uniform.vector) %33 = call <vscale x 2 x i64> @llvm.aarch64.sve.compact.nxv2i64(<vscale x 2 ... x i1 > %31, <vscale x 2 x i64 > %remaining.vector) %34 = call i64 @llvm.aarch64.sve.cntp.nxv2i1(<vscale x 2 x i1> %27, <vscale ... $\times 2 \times i1 > \%20$ %35 = call <vscale x 2 x i1> @llvm.aarch64.sve.whilelt.nxv2i1.i64(i64 0, i64 ... %34) %36 = call <vscale x 2 x i64> @llvm.aarch64.sve.splice.nxv2i64(<vscale x 2 x .. i1> %35, <vscale x 2 x i64> %28, <vscale x 2 x i64> %29) %37 = call i64 @llvm.aarch64.sve.cntp.nxv2i1(<vscale x 2 x i1> %27, <vscale ... $\times 2 \times i1 > \%20$ %38 = call <vscale x 2 x i1> @llvm.aarch64.sve.whilelt.nxv2i1.i64(i64 0, i64 .. %37) %39 = call <vscale x 2 x i64> @llvm.aarch64.sve.splice.nxv2i64(<vscale x 2 x ... i1> %38, <vscale x 2 x i64> %29, <vscale x 2 x i64> %33) %40 = call i64 @llvm.aarch64.sve.cntp.nxv2i1(<vscale x 2 x i1> %27, <vscale ... x 2 x i1 > %30%41 = call <vscale x 2 x i1> @llvm.aarch64.sve.whilelt.nxv2i1.i64(i64 0, i64 ... %40) %42 = call <vscale x 2 x i64> @llvm.aarch64.sve.sel.nxv2i64(<vscale x 2 x ... i1> %41, <vscale x 2 x i64> \(\tilde{\sigma} \) 32, <vscale x 2 x i64> %39) %43 = xor < vscale x 2 x i1 > %41, shufflevector (< vscale x 2 x i1 > %41) ... insertelement (<vscale x 2 x i1> poison, i1 true, i32 0), <vscale x 2 x i1> ... poison, <vscale x 2 x i32> zeroinitializer) %44 = call i64 @llvm.aarch64.sve.cntp.nxv2i1(<vscale x 2 x i1> %27, <vscale ... x 2 x i1 > %41) %45 = sub i64 %37, %40%46 = add i64 %45, %44 %47 = call <vscale x 2 x i1> @llvm.aarch64.sve.whilelt.nxv2i1.i64(i64 0, i64 ... %46) %48 = xor < vscale x 2 x i1 > %47, shufflevector (< vscale x 2 x i1 > %48) ... insertelement (<vscale x 2 x i1> poison, i1 true, i32 0), <vscale x 2 x i1> ... poison, <vscale x 2 x i32> zeroinitializer) %49 = and < vscale x 2 x i1 > %43, %47%50 = and < vscale x 2 x i1 > %47, %48%51 = or < vscale x 2 x i1 > %49, %50br label %alc.applied alc.applied: linearized: br label %new.latch br label %new.latch new.latch: %52 = add i64 %9, %24 %53 = add <vscale x 2 x i64> %25, %stepVector.update.values %54 = add <vscale x 2 x i64> %26, %stepVector.update.values br i1 true, label %alc.header, label %middel.block middel.block: %condition = icmp eq i64 %10, 0 br i1 %condition, label %for.cond.cleanup4, label ... %Preheader.for.remaining.iterations Preheader.for.remaining.iterations: %7 = phi i64 [0, %for.cond2.preheader], [%52, %middel.block] br label %for.body5 for.body5: %indvars.iv = phi i64 [%indvars.iv.next, %for.inc], [%7, ... %Preheader.for.remaining.iterations] call void @llvm.dbg.value(metadata i64 %indvars.iv, metadata !88, metadata ...!DIExpression()), !dbg!100 %arrayidx = getelementptr inbounds [8192 x i32], ptr @a, i64 0, i64 ... %indvars.iv, !dbg !116 %3 = load i32, ptr %arrayidx, align 4, !dbg !116, !tbaa !120 %arrayidx7 = getelementptr inbounds [8192 x i32], ptr @b, i64 0, i64 ... %indvars.iv, !dbg !124 %4 = load i32, ptr %arrayidx7, align 4, !dbg !124, !tbaa !120 %cmp8 = icmp sgt i32 %3, %4, !dbg !125 br i1 %cmp8, label %if.then, label %for.inc, !dbg !126 if.then: %arrayidx14 = getelementptr inbounds [8192 x i32], ptr @d, i64 0, i64 ... %indvars.iv, !dbg !127 %5 = load i32, ptr %arrayidx14, align 4, !dbg !127, !tbaa !120 %mul = mul nsw i32 %5, %4, !dbg !129 %sub = sub nsw i32 %3, %mul, !dbg !130 call void @llvm.dbg.value(metadata i32 %sub, metadata !85, metadata ...!DIExpression()),!dbg!92 %arrayidx16 = getelementptr inbounds [8192 x i32], ptr @c, i64 0, i64 ... %inďvars.iv, !ďbg !131 %6 = load i32, ptr %arrayidx16, align 4, !dbg !132, !tbaa !120 %add = add nsw i32 %sub, %6, !dbg !132 store i32 %add, ptr %arrayidx16, align 4, !dbg !132, !tbaa !120 store i32 %sub, ptr %arrayidx, align 4, !dbg !133, !tbaa !120 br label %for.inc, !dbg !134 for.inc: %indvars.iv.next = add nuw nsw i64 %indvars.iv, 1, !dbg !135 call void @llvm.dbg.value(metadata i64 %indvars.iv.next, metadata !88, ... metadata !DIExpression()), !dbg !100 %exitcond.not = icmp eq i64 %indvars.iv.next, 8192, !dbg !136 br i1 %exitcond.not, label %for.cond.cleanup4, label %for.body5, !dbg !101, ... !llvm.loop !137 for.cond.cleanup4: %call19 = tail call i32 @dummy(ptr noundef nonnull @a, ptr noundef nonnull ... @b, ptr noundef nonnull @c, ptr noundef nonnull @d, ptr noundef nonnull @e, ... ptr noundef nonnull @aa, ptr noundef nonnull @bb, ptr noundef nonnull @cc, ... i32 noundef 0) #8, !dbg !109 %inc21 = add nuw nsw i32 %nl.038, 1, !dbg !110 call void @llvm.dbg.value(metadata i32 %inc21, metadata !86, metadata ... !DIExpression()), !dbg !98 %exitcond40.not = icmp eq i32 %inc21, 10, !dbg !111 br i1 %exitcond40.not, label %for.cond.cleanup, label %for.cond2.preheader, ...!dbg!99,!llvm.loop!112 F for.cond.cleanup: tail call void asm sideeffect ".inst 0x2520e040", ""() #8, !dbg !102, ...!srcloc!104 %t2 = getelementptr inbounds %struct.args t, ptr %func args, i64 0, i32 1, ... !dbg !105 %call23 = tail call i32 @gettimeofday(ptr noundef nonnull %t2, ptr noundef ... null) #8, !dbg !106 %call24 = tail call i32 @calc checksum(ptr noundef nonnull @ func .s253) ... #8, !dbg !107

entry:

CFG for 's253' function

ret i32 %call24, !dbg !108